COLLEGE CODE: 9233

COLLEGE NAME :GOVERNMENT COLLAGE OF ENGINNERING BODINAYAKANUR

DEPARTMENT:COMPUTER SCIENCE AND ENGINNERING

STUDENT NM-ID: 8AE480233C809F1A2D54F980ABF4887C

ROLL NO: 923323104305

DATE: 25.09.2025

Completed the project named as Phase__1

TECHNOLOGY PROJECT NAME :REAL TIME CHAT BOT

SUBMITTED BY,

NAME :THIRUPATHI P

MOBILE NO: 9345110398

Phase 1: Problem Understanding & Requirements (Deadline: TBD)

Problem Statement

Current customer support systems are often delayed or require human intervention, leading to poor response times. A real-time chat bot can handle user queries instantly, provide relevant information, and escalate to a human when necessary, reducing support costs and improving user experience.

Users & Stakeholders

- End Users : Website/app visitors, customers needing instant answers.
- Admins/Business Owners : Want analytics, chat logs, and efficient handling of repetitive queries.
- Technical Stakeholders : Developers integrating chatbot into frontend and backend REST API system.

User Stories

- As a user , I want to send and receive messages instantly, so I can get quick answers.
- As an admin , I want to monitor chat history, so I can review customer issues.
- As a system , I want to detect when a query cannot be answered and escalate it.
- As a developer , I want to expose REST API endpoints for chat history, user sessions, and analytics.

MVP Features

- Real time messaging between user and chatbot.
- Node.js REST API for managing chat sessions and message logs.
- Frontend interface with chat UI (built with React/Vue/Angular).
- Predefined responses from chatbot for FAQs.
- Escalation mechanism to route queries to a human admin.

Wireframes / API Endpoint List

- Chat window UI with message bubbles (user vs bot).
- Admin dashboard for chat logs.

API Endpoint List

- POST /api/chat/send → Send message from user.
- GET /api/chat/:sessionId → Fetch chat history.
- POST /api/chat/respond → Bot generates response.
- POST /api/chat/escalate → Escalate chat to admin.
- GET /api/admin/logs → Retrieve all user chat logs.

Acceptance Criteria

- User is able to send a message and receive a bot reply instantly (<1 second latency).
- All messages are stored in the database and retrievable via API.
- If query is not matched with a predefined response, the system escalates to admin.
- Chat UI is functional on desktop and mobile.
- Admins can view and search previous chat logs.